

Thema: Künstliche Intelligenz

KI/AI: Vortäuschen von intelligentem Verhalten

Turing 1950: "Are there imaginable digital computers that would do well in the imitation game?"

Turing Test: Mensch versucht Computer und anderen Menschen durch Fragen zu unterscheiden.

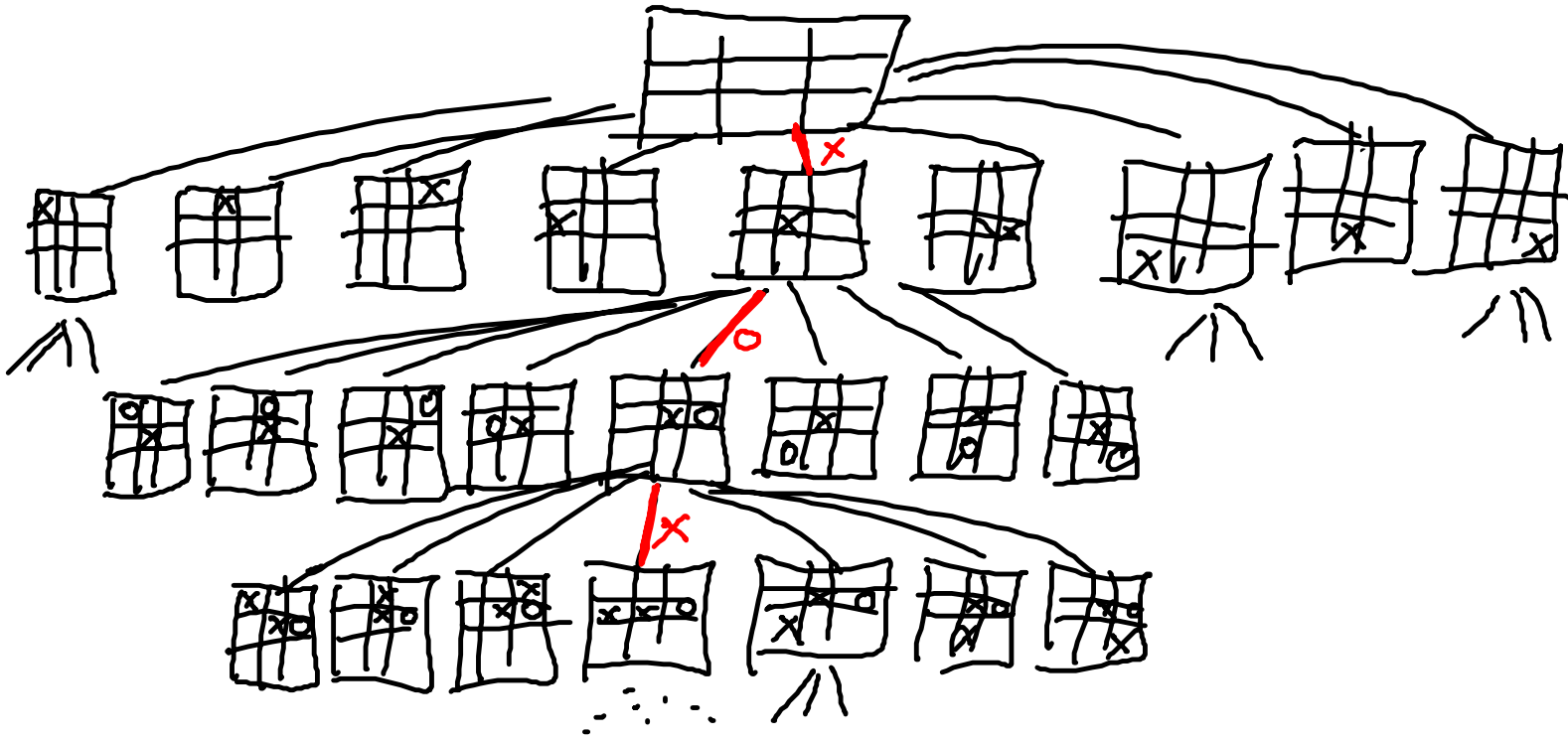
~~Passes~~ wildly : was steht hier : Captcha = Completely Automated Public Turing Test to tell Computers and Humans Apart

KI: - Komplexe Aufgaben ohne opt. Algorithmen
- meist: Anwendung best. Regeln, Training von Parametern an Daten,
aufwendige Suche aller/mögl. Möglichkeiten

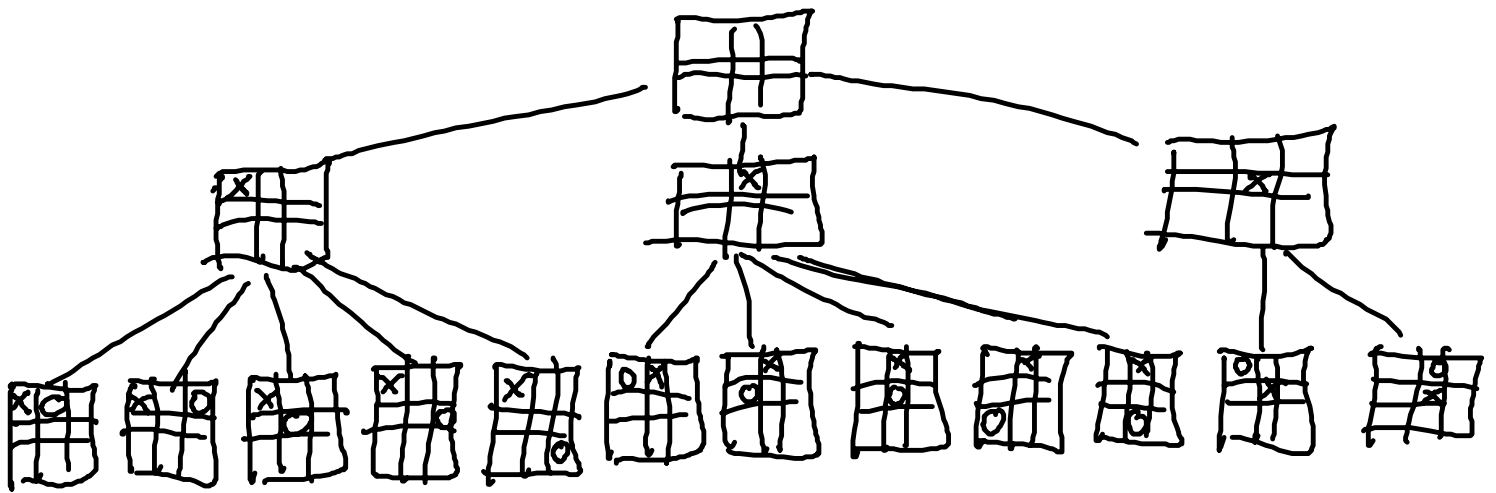
Spiele:	deterministische	(teilweise) zufällig
vollst. Information	Schach, Go, Dame, Mühle, Hive	Backgammon, Mensch gegen dich nicht
unvollst. Information	Schiffe versenken, Stratego,	Skat, Scrabble, Bridge

Beispiel: Tic Tac Toe

0	0	x
x	x	0
0	x	x



Tic Tac Toe (Symmetrien beachten)



Tiefe 2: 12 Knoten

ohne Sym. reduktion: $3 \cdot 8 = 72$ Knoten

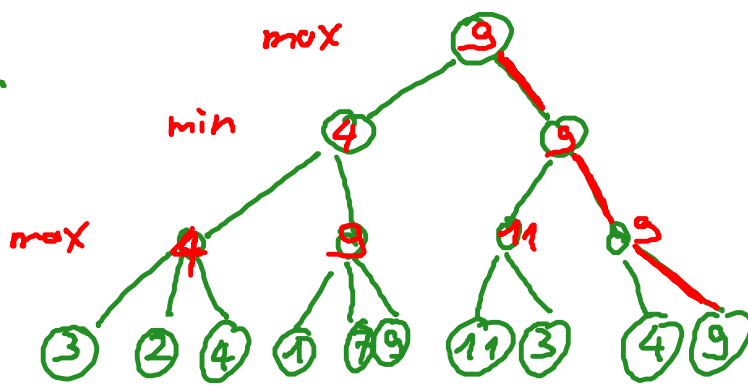
Tic Tac Toe hat $\leq 9!$ Spielstände = 362.880

bei Sym. Redukt: 765

Daf: Spielbaum:

- Wurzel: Ausgangszustand
- Knoten: nächster Spielstand
- Kante: gültigem Zug
- Blätter: Endzustand
- Bewertung der Blätter: z.B. +1 Sieg oder Spielstand
-1 Niederlage
0 Unentschieden.

Beispiel:

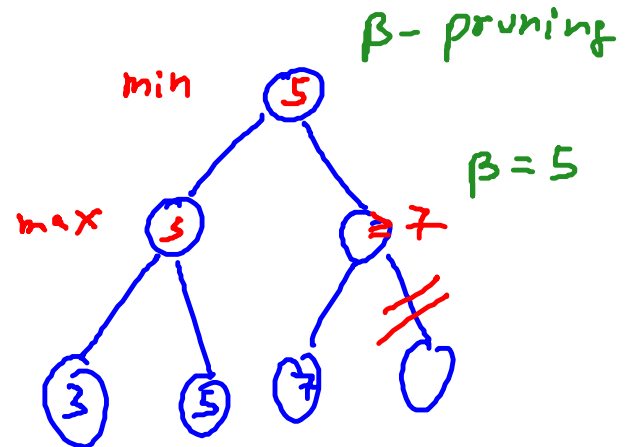
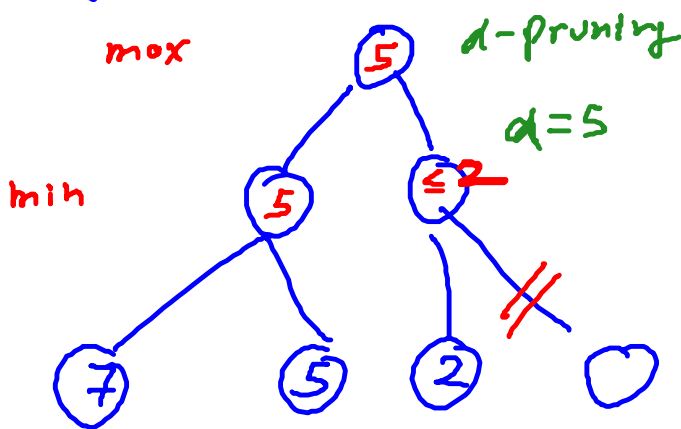


- 2 Spieler, abwechselnd spielend
- 1. Spieler: maximiere Endstand
- 2. Spieler: min. Endstand

Algo: Min Max: abwechselnd min/max wählen

- optimale Spielstrategie bei perfekt spielendem Gegner
- Spiel zu Unentschieden/Gewinn führbar (sofern möglich)
- Beweis Damp-programm (2007): Verliert nie \Rightarrow 2 perfekt spielen immer Unentschieden
- Mühle: 1,8 $\cdot 10^{10}$ Stellungen (elin. Rotationen, Spiegelungen & Vertauschung) \rightarrow 2 perf. Spieler spielen Unentschieden
- Schach: # Stellungen ungefähr $2,28 \cdot 10^{46}$ und 10^{120} Spielverläufe ($\sim 10^{80}$ Atome im Universum)

Verbesserung von Min Max:



\rightarrow Teilbäume abschneiden (nicht auswerten)

genannt α - β Pruning:

Idee: einige Teilbäume spielen bei Auswertung keine Rolle \Rightarrow Abschneiden & nicht weiter auswerten

- Alpha: min. Wert für Spieler Max
- Beta: max. Wert für Spieler Min

Pseudocode: start: AlphaBeta(wurzel, $-\infty, \infty$)

ohne Rot: Min Max Algo

AlphaBeta(node, α , β)

Input: node ist der aktuell zu behandelnde Knoten / aktueller Zustand

die Werte α , β geben das Fenster für den Knoten an

Output: Wert α für Knoten node, falls dieser ein Max-Knoten ist und β sonst

IF (node ist ein Blatt) **THEN**

RETURN Auswertung zum Knoten node

ENDIF

IF (node ist ein Max-Knoten) **THEN**

erzeuge alle gültigen Nachfolgezüge

WHILE (es gibt noch einen unbehandelten gültigen Nachfolgezug move) **DO**

erzeuge Knoten child durch Anwendung von move auf node

a:=AlphaBeta(child, α , β)

nimm Anwendung von move auf node zurück

IF ($a \geq \beta$) **THEN**

RETURN β

ENDIF

IF ($a > \alpha$)**THEN**

$\alpha := a$

ENDIF

ENDWHILE

RETURN α

ELSE //Min-Knoten

erzeuge alle gültigen Nachfolgezüge

WHILE (es gibt noch einen unbehandelten gültigen Nachfolgezug move) **DO**

erzeuge Knoten child durch Anwendung von move auf node

b:=AlphaBeta(child, α , β)

nimm Anwendung von move auf node zurück

IF ($b \leq \alpha$) **THEN**

RETURN α

ENDIF

IF ($b < \beta$)**THEN**

$\beta := b$

ENDIF

ENDWHILE

RETURN β

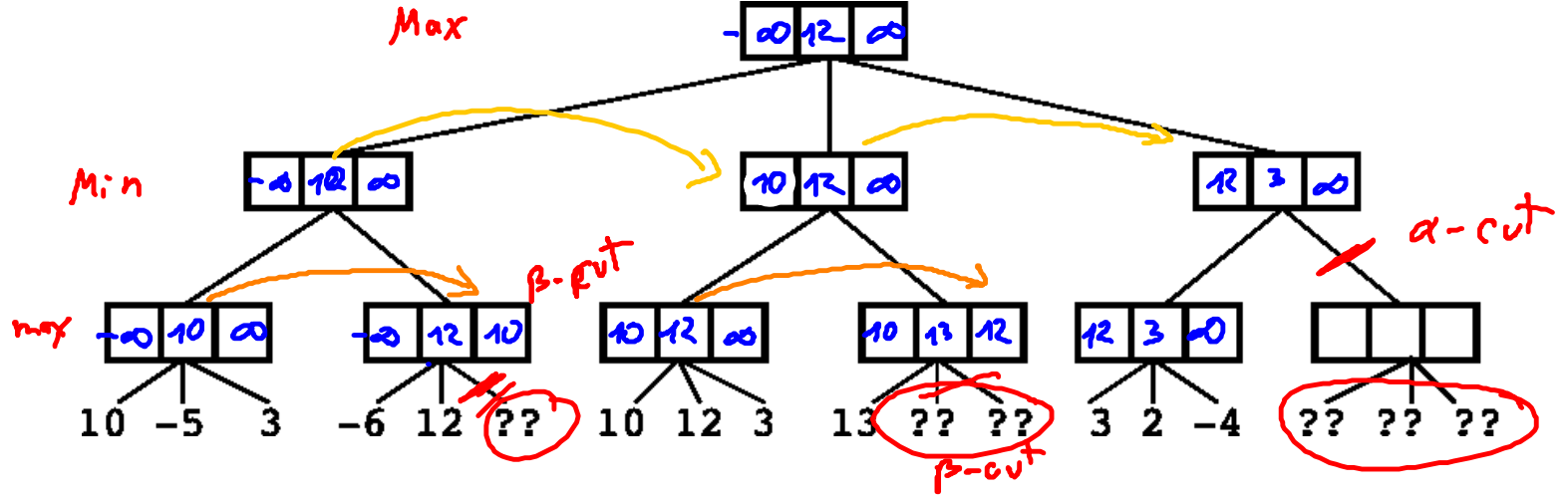
ENDIF

} pruning

} pruning

Beispiel:

α weiß β



Baumgröße: b = Anzahl der Kinder eines Knotens
 m = Spiellänge / Baumhöhe

\Rightarrow Baumgröße $\in O(b^m)$

- Best Case d/B Pruning: $O(b^{m/2})$ (gleicher Laufzeit \Rightarrow doppelt so weit in Zukunft + sehen)
- α - β Pruning: abhängig von auswertungs Reihenfolge
- immer noch nicht gut genug um komplexe Spiele auszuwerten

Heuristiken: - anstatt völlig lösen, nur best. Tiefe berechnen und Stellung bewerten (z.B. Figuren im Spiel, geschl. Figuren)

Probleme: Spiele wie Go: - viele mögliche Züge (\rightarrow breiter Spielbaum)
 - langes Spiel: ≥ 150 Züge (Hoher Baum)
 - Stellungsbewertung unklar
 \rightarrow Spieler oft uneinig

Weitergehende Ideen:

- Eröffnungsdatenbank
- feste Regeln \rightarrow Siege herbeiführen/Niederlage abwenden/Dame schlagen ..
- Monte Carlo Spielbaum Bewertung
- dyn. Baum Tiefe (Zeit, stellungabhängig)
- Stellungen nicht doppelt bewerten \rightarrow Abspeichern + Zugreifen
- Symmetrien (Rotationen, Spiegelungen, Vertauschungen) beachten
- Züge sortieren und auswerten